IFC - Campus Videira
Relatório Descritivo dos Experimentos com RabbitMQ e Node.js
Thiago Trzcinski

Introdução:
Este relatório descreve uma série de experimentos realizados utilizando RabbitMQ, um sistema de mensagens de código aberto, e Node.js, uma plataforma de desenvolvimento de aplicativos em JavaScript. Os experimentos visam demonstrar diferentes padrões de comunicação entre processos usando RabbitMQ, com exemplos práticos implementados em Node.js.

Linguagem de Programação Escolhida:
Para implementar os exemplos, foi escolhido JavaScript, utilizando o ambiente de tempo de execução Node.js. Essa escolha foi motivada pela familiaridade com Node.js no desenvolvimento de aplicativos web e pela disponibilidade da biblioteca `amqplib`, que oferece suporte para interação com RabbitMQ.

Código Desenvolvido:
O código desenvolvido consiste em uma série de scripts JavaScript que demonstram diferentes padrões de troca de mensagens usando RabbitMQ. Cada tutorial possui seu próprio conjunto de scripts, conforme descrito a seguir:

1. Tutorial Hello World!:
- send.js: um script que envia uma mensagem simples para uma fila no RabbitMQ.
- receive.js: um script que recebe mensagens da fila e as imprime no console.

2. Tutorial Work Queues:
- new_task.js: Um script que simula uma tarefa demorada e envia uma mensagem para uma fila no RabbitMQ.
- worker.js: Um script que simula um trabalhador que recebe e processa tarefas da fila.

3. Tutorial Publish/Subscribe:
- receive_logs.js: Um script que se inscreve em um canal de mensagens e imprime as mensagens recebidas no console.
- emit_log.js: Um script que publica uma mensagem em um canal de mensagens.

4. Tutorial Routing:
- receive_logs_direct.js: Um script que se inscreve em um canal específico de mensagens baseado em um nível de severidade e imprime as mensagens recebidas no console.
- emit_log_direct.js: Um script que publica uma mensagem em um canal específico de mensagens com uma chave de roteamento.

5. Tutorial Topics:
- receive_logs_topic.js: Um script que se inscreve em um canal específico de mensagens com base em padrões de roteamento e imprime as mensagens recebidas no console.
- emit_log_topic.js: Um script que publica uma mensagem em um canal específico de mensagens com uma chave de roteamento baseada em padrões.

6. Tutorial RPC(Remote Procedure Call):

- rpc_server.js: Um script que implementa um servidor RPC.
- rpc_client.js: Um script que implementa um cliente RPC e envia uma solicitação para o servidor.

Análise dos Resultados Obtidos:

Os resultados obtidos demonstram com sucesso diferentes padrões de comunicação entre processos usando RabbitMQ. Cada tutorial exemplifica um cenário específico de uso, desde mensagens simples até chamadas de procedimento remoto. Os exemplos fornecem uma base para entender como RabbitMQ pode ser integrado a aplicativos Node.js para implementar sistemas distribuídos e escaláveis.

Dificuldades Encontradas:

Durante a implementação dos experimentos, algumas dificuldades foram encontradas, incluindo:

- Configuração inicial do ambiente RabbitMQ e Node.js.
- Entendimento dos conceitos de filas, trocas e chaves de roteamento em RabbitMQ.
- Implementação correta dos padrões de comunicação, especialmente no que diz respeito à sincronização entre produtores e consumidores.